COMPOSITE KEYSTORE FACILITY APPARATUS AND METHOD THEREFOR

TECHNICAL FIELD

The present invention relates in general to data processing systems, and in particular, to the management of databases of cipher keys for protecting sensitive transactions in the data processing system.

BACKGROUND INFORMATION

Transactions between data processing systems that require the transfer of sensitive information are ubiquitous in the modern data processing environment. For example, such transactions may include electronic commerce retail transactions by individuals and businesses via the Internet, electronic funds transfers between financial institutions, or the communication of proprietary information between multiple installations of an enterprise. Protection of the sensitive information may be implemented by encrypting, or enciphering, the sensitive information, thereby shielding the sensitive information from unauthorized access. Modern cryptographic schemes rely on encryption algorithms that are publicly known, and rely on a secret encryption key (hereinafter simply "key"). *See e.g.,* BRUCE SCHNEIER APPLIED CRYPTOGRAPHY SECOND

EDITION 31-32 (1997). Enciphering an information set to be protected entails mathematically combining a key and the data constituting the information to be protected in accordance with the encryption algorithm. *See e.g. Id.* at 4-5. The recipient of the data recovers the information (the so-called plaintext) by mathematically combining a decryption key with the enciphered data. *See e.g. Id.* So-called symmetric encryption algorithms use the same key for enciphering the plaintext to generate the encrypted data and for decrypting the enciphered data to recover the plaintext. *See e.g. Id.* Asymmetric key encryption algorithm, which may also be referred to as public key algorithms, encipher the data to be protected with a first key, called the public key, and recover the plaintext with a second key, referred to as the private key. *See e.g. Id.*

Referring to FIGURE 1, there is illustrated therein a schematic transaction between two users, A and B, illustrating the operation of a public key encryption scheme for protecting the sensitive information in the transaction. Note that one or both of users A and B need not be human users, users may also refer to data processing systems automatically implementing the transaction without human intervention. For example, a data processing system in a main office of a financial institution may automatically gather financial records from branch institutions at the close of business, without human intervention, and for security purposes, the transfer of such information may be protected by enciphering the data. Similarly, in electronics funds transferred between financial institutions, which need not necessarily be branches of the same enterprise, may similarly be effected. Suppose, for example, in FIGURE 1, user A needs to communicate sensitive information to user B. Encryption of the data is to be performed using a public key

encryption algorithm, which is known to both user A and user B. Such algorithms are known in the data processing art. *See e.g. Id* at 466-67; 513-14 (discussing the RSA and Diffie-Hellman algorithms.) Each user has an associated private key (user A's, 102 and user B's 106) and a public key (user A's 104 and user B's 106). In accordance with the principles of public key cryptosystems, user A issues a request 110 to user B (as would be understood by an artisan of ordinary skill in the art, the data processing system associated with user A issues a request of the data processing system associated with user B) for user B's public key 108. User B sends a reply 112 to user A containing user B's public key 108. User A encrypts user A's message with user B's public key 108 and sends the encrypted message 114 to user B. User B then decrypts the message 114 using user B's private key 106 which is known only to user B. Note that user A's encrypted message 114 cannot be decrypted using B's public key. (FIGURE 1 is meant to be illustrative of public key systems. More typically the public key algorithms is used to encrypt a secret symmetric key which is used with a corresponding symmetric key algorithm to encrypt the message, and the ciphertext and encrypted secret key are both sent to the recipient who first decrypts the secret key and uses it to recover the plaintext.)

However, user A cannot, without more, be certain that user B's public key reply message 112 has not been compromised in flight where, for example, a third-party substitutes its public key for user B's public key and not the public key of another. In other words, user A, without more, cannot be certain that the public key message 112 that it receives actually contains user B's public key. To authenticate user B's private key, the private key may be digitally signed by a party trusted by user A. (Authentication using

digital signing is discussed, for example, in BRUCE SCHNEIER, APPLIED CRYPTOGRAPHY SECOND EDITION 34-41 (1997).) For the purposes herein, a file or data structure including a private key and associated digital signature (and possibly other information as discussed further below) may be referred to as a certificate. Trusted parties signing certificates may be referred to as certificate authorities (CAs)(Authentication of certificates may be provided by third-party commercial CAs. One such CA is Verisign). Thus, each user, such as user A and B in FIGURE 1 maintain a database (often referred to as a keystore) of certificates which includes a certificate associated with the users' respective public key, as well as certificates associated with one or more certificate authorities, which, as will be discussed further herein below, may be necessary to create a "chain of authorities." Typically, security enabled applications, web browsers, for example, usually include a keystore that includes certificates associated with one or more certificate authorities that are trusted parties of the provider of the web browser, or other application. Consequently, to accommodate authenticated key exchange, each user maintains a keystore 114 that includes a set of certificates needed recognize and resolve (or possibly, to generate) a chain of authorities. Such a keystore is schematically illustrated in FIGURE 1B. In addition to the private key information 102, a portion of the keystore 116 contains the public information which includes a set of *n* of certificates 118, labeled CA1, CA2, CAn. Consequently, each user, maintains a set, or database of keystores associated with each security enabled application in the user's data processing system. This can create a significant administrative workload to maintain each user's keystore. For example, certificates may have a finite life time, and one or more

certificates may expire without the users knowledge, and additionally, individual users may not have the expertise necessary to update expired certificates. As a consequence, a chain of authentication may be broken, and the user unable to effect secure communications. Thus, there is a need in the art for apparatus and methods to centralize the management of certificate databases.

SUMMARY OF THE INVENTION

The aforementioned needs are addressed by the present invention. Accordingly, there is provided, in a first form, a keystore method. The method includes reprieving one or more certificates from a local database. It is determined if said any of said one or more certificates preexists in a preselected portion of a distributed database, and nonpreexisting certificates of said one or more certificates are stored in the preselected portion of the distributed database.

There is also provided, in a second form, a computer program product in a tangible storage medium. The program product includes a program of instructions for performing the process of

retrieving one or more certificates from a first local database. The instructions further determine if the any of the one or more certificates preexists in a preselected portion of a distributed database, and store nonpreexisting certificates of the one or more certificates in the preselected portion of the distributed database.

Additionally, there is provided, in a third form, a data processing system. The system contains circuitry operable for retrieving one or more certificates from a first local database. Also included is circuitry operable for determining if the any of the one or more certificates preexists in a preselected portion of a distributed database, and circuitry operable for storing nonpreexisting certificates of the one or more certificates in the preselected portion of the distributed database.

The foregoing has outlined rather broadly the features and technical advantages of the present invention in order that the detailed description of the invention that follows may be better understood. Additional features and advantages of the invention will be described hereinafter which form the subject of the claims of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the present invention, and the advantages thereof, reference is now made to the following descriptions taken in conjunction with the accompanying drawings, in which:

FIGURE 1A schematically illustrates an encryption key exchange which may be used in an embodiment of the present invention;

FIGURE 1B schematically illustrates a keystore structure which may be used in an embodiment of the present invention;

FIGURE 2 illustrates, in block diagram form, a data processing system implemented in accordance with an embodiment of the present invention;

FIGURE 3 illustrates, in flow chart form, a composite keystore methodology in accordance with an embodiment of the present invention;

FIGURE 4 schematically illustrates a key certificate which may be used in an embodiment of the present invention;

FIGURE 5 illustrates, in flow chart form, a data transfer methodology in accordance with another embodiment of the present invention; and

FIGURE 6 illustrates, in flow chart form, a composite keystore methodology in accordance with an alternative embodiment of the present invention.

DETAILED DESCRIPTION

The present invention provides a system and method for aggregating public, non-personal authenticated encryption keys (contained in certificates, which may be X.509 certificates, described further herein below) in a database (which may be referred to as a keystore). The certificates may include certificates for individual users, root certificates for certificate authorities (CAs) and additional certificates, which may be required to generate a chain of authorities. The aggregated keystore may be a hierarchical or multilevel keystore, which may be associated with an organizational structure of an enterprise. A first-level or "local" keystore may include personal certificates and one or more higher-level, or "organizational" keystores may include root certificates for CAs considered trusted by the enterprise or corresponding organization. Additionally, a centralized management of certificates may be provided, whereby for example, the expiration or revocation of the certificates may be tracked, and expired or revoked certificates may be refreshed.

In the following description, numerous specific details are set forth such as specific application program interfaces (APIs) to provide a thorough understanding of the present invention. However, it will be obvious to those skilled in the art that the present invention may be practiced without such specific details. In other instances, well-known circuits have been shown in block diagram form in order not to obscure the present invention in unnecessary detail. For the most part, details concerning timing considerations and the like have been omitted in as much as such details are not

necessary to obtain a complete understanding of the present invention and are within the skills of persons of ordinary skill in the relevant art.

Refer now to the drawings wherein depicted elements are not necessarily shown to scale and wherein like or similar elements are designated by the same reference numeral through the several views.

Referring first to FIGURE 2, an example is shown of a data processing system 200 which may be used for the invention. The system has a central processing unit (CPU) 210, which is coupled to various other components by system bus 212. Read only memory ("ROM") 216 is coupled to the system bus 212 and includes a basic input/output system ("BIOS") that controls certain basic functions of the data processing system 200. Random access memory ("RAM") 214, I/O adapter 218, and communications adapter 234 are also coupled to the system bus 212. I/O adapter 218 may be a small computer system interface ("SCSI") adapter that communicates with a disk storage device 220. Communications adapter 234 interconnects bus 212 with an outside network enabling the data processing system to communicate with other such systems. Input/Output devices are also connected to system bus 212 via user interface adapter 222 and display adapter 236. Keyboard 224, track ball 232, mouse 226 and speakers 228 are all interconnected to bus 212 via user interface adapter 222. Display monitor 238 is connected to system bus 212 by display adapter 236. In this manner, a user is capable of inputting to the system throughout the keyboard 224, trackball 232 or mouse 226.

Preferred implementations of the invention include implementations as a computer system programmed to execute the method or methods described herein, and as a computer program product. According to the computer system implementation, sets of instructions for executing the method or methods are resident in the random access memory 214 of one or more computer systems configured generally as described above. Until required by the computer system, the set of instructions may be stored as a computer program product in another computer memory, for example, in disk drive 220 (which may include a removable memory such as an optical disk or floppy disk for eventual use in the disk drive 220). Further, the computer program product can also be stored at another computer and transmitted when desired to the user's work station by a network or by an external network such as the Internet. One skilled in the art would appreciate that the physical storage of the sets of instructions physically changes the medium upon which it is stored so that the medium carries computer readable information. The change may be electrical, magnetic, chemical, biological, or some other physical change. While it is convenient to describe the invention in terms of instructions, symbols, characters, or the like, the reader should remember that all of these and similar terms should be associated with the appropriate physical elements.

Note that the invention may describe terms such as comparing, validating, selecting, identifying, or other terms that could be associated with a human operator. However, for at least a number of the operations described herein which form part of at least one of the embodiments, no action by a human operator is desirable. The operations

described are, in large part, machine operations processing electrical signals to generate other electrical signals.

Refer now to FIGURE 3 illustrating, in flow chart form, methodology 300 for aggregating public, nonpersonal keystore information into a composite keystore, which may interchangeably be referred to as an aggregate keystore or distributed keystore. (The flow charts provided herein are not necessarily indicative of the serialization of the operations being performed in an embodiment of the present invention. Many of the steps disclosed in the flow charts may actually be performed in parallel. The flow chart is meant to designate those considerations that may be performed to produce the operation available on distributed keystores.) In an embodiment of the present invention, the composite key store may be created across an entire enterprise. Alternatively, a multi-level composite keystore may be implemented. A multi-level keystore (which may also be referred to as a hierarchical, or organizational keystore) aggregates certificates into keystore groups, which may be structured to parallel the organizational scheme of an enterprise, or other preselected keystore structure. In step 304, certificates are retrieved from the local keystores for users on the system. Methodology 300 aggregates a keystore for each user by retrieving keystores on a user-by-user basis, as will be described in conjunction with steps 306-316.

In step 306, it is determined if certificates retrieved from a users local keystore in step 304 are expired, or revoked. A certificate that has been compromised, for example, may be revoked by a CA. (Certificates that are expired or revoked may collectively be referred to as invalid certificates.) Referring to FIGURE 4, there is

schematically indicated therein the structure of a certificate 400 which may be used within the present invention. Certificate 400 may be a certificate in accordance with X.509 protocols. (X.509 refers to the international standard for digital certificates promulgated by the International Telecommunications Union - Telecommunications Standards Section (ITU-TSS).) Certificate 400 includes eight fields 402-416. Version field 402 identifies the certificate format. (For example, three versions of X.509 certificates have been specified.) Serial number field 404 contains a unique serial number assigned by the CA. Algorithm identifier field 406 identifies the algorithm used to sign the certificate. Field 406 may also include subfields containing any necessary parameters associated with the particular algorithm used to sign the certificate. (As previously discussed, digital signal algorithms are available in the data processing art, however, an artisan of ordinary skill would recognize that the particular digital signal algorithm used to sign a particular certificate is not germane to the present invention; the principles of the present invention are independent of the particular digital signal algorithm. Issuer field 408 includes the name of the CA issuing the certificate. Validity period field 410 includes a pair of dates delimiting the time period during which the certificate is valid. Subject field 412 includes the identity of the party whose public key is contained in the certificate, and which is being authenticated by the certificate. Public key field 414 includes the public key of the subject identified in field 412, along with additional information, such as the public key algorithm associated with the public key, and any necessary parameters required by the algorithm. Signature field 416 includes the digital signature of the CA issuing the certificate, which issuer appears in field 408.

Thus, in a certificate in accordance with the X.509 specifications, step 306 may be performed by examining the values in validity period field 410.

Returning to FIGURE 3, if in step 306, expired or revoked certificates are found, the expired or revoked certificates are entered in a list thereof, in step 308. In step 310, it is determined if the certificates retrieved in step 304 are to be aggregated in a multi-level keystore. If so, the nonduplicative, unexpired unrevoked certificates are added to the corresponding keystore group in step 312. Otherwise, the nonduplicative, unexpired unrevoked certificates are added to the enterprise-wide keystore, step 314. Nonduplicative certificates refers to certificates retrieved in step 304 that do not otherwise exist in the keystore. Because certificates are aggregated over multiple users, and as previously discussed, certain applications (web browsers, for example) may install an embedded keystore of certificates included by the provider of the application software, each user having a local copy of such software may also replicate the common certificates. Such certificates, for example, need not be aggregated in multiple instances.

In step 316, it is determined if the local keystores of all users have been aggregated. If not, methodology 300 returns to step 304 to retrieve the certificates for a next user whose local keystores are to be incorporated in the aggregated keystore. If, in step 316, all user's local keystores have been aggregated, step 316 proceeds by the "Yes" branch and in step 318 it is determined if the expired/revoked certificate list, compiled in step 308 is empty. If the list is not empty, in step 319, it is determined if the expired/revoked certificates are to be refreshed. This may be set in accordance with a predetermined security policy. If so, in step 320, a new certificate request is sent to the

issuer and the certificate deleted from the list generated in step 308. Recall that in a certificate in accordance with the X.509 specification, the CA issuing the certificate may be determined by examining the issuer field, field 408 in FIGURE 4. When a requested certificate returns, it is added to the keystore, step 322. As would be understood by an artisan of ordinary skill, the return of a certificate from a CA may occur after an elapse of a relatively long period of time, on the order of a day.

Returning to step 318, if no expired/revoked certificates were found, the list of expired/revoked certificates is empty, and step 318 proceeds by the "Yes" branch. In step 324 it is determined if user updates are permitted. User updates may be permitted or denied in accordance with a predetermined policy. If, such a policy permits users to update certificates in the distributed keystore, then in step 326 user updates are received in step 326. Additionally, methodology 300 may automatically update the distributed keystore by, for example, tracking the expiration dates of certificates in the distributed keystore. Also, a distributed keystore may be updated by, for example, adding additional certificates in response to system administrator input. It would be understood by an artisan of ordinary skill that these events for updating the distributed keystore are exemplary, and that other events would be within the spirit and scope of the present invention. If, in step 328 the keystore is to be updated, in step 330 the certificate is requested, and methodology 300 proceeds to step 322. Otherwise, methodology 300 terminates step 332.

Refer now to FIGURE 6 illustrating, in flow chart form, a methodology 600 for generating a composite keystore in accordance with an alternative embodiment of the

present invention. Note that the composite keystore may be a logical object, and process 600 need not implicate the physical transfer of certificates. Process 600 composes the keystore in steps 604-616. While there are more keystore sources from which to populate the composite keystore, step 604 loops over steps 606-616. A keystore source may be represented by a location associated with a Uniform Resource Locator (URL), as discussed herein below. Each such keystore may be accessed for retrieving certificates using the URL. While there are more keystore sources, step 604, in step 606, all the certificates are retrieved from a current source. While, there are certificates to process from the current keystore source, step 608, it is determined in step 610 if a current certificate is already in the composite keystore. If not, the certificate is added to the composite, step 612, and methodology 600 returns to step 608.

If, the certificate already exists in the composite keystore, in step 614 it is determined if the certificate being processed is "better" then the corresponding certificate in the composite keystore. A keystore may be "better" in accordance with a set of predetermined criteria. If the certificate being processed is unexpired while the preexisting corresponding certificate in the keystore is expired, or if the certificate being processed is unrevoked while the corresponding certificate in the composite keystore is revoked, or in accordance with a predetermined policy-based decision. An exemplary policy-based decision may determine that a version of the certificate being processed that is higher in the composite hierarchy is "better", or alternatively, "worse", as a matter of a predetermined security policy. It would be understood by an artisan of ordinary skill that the aforementioned criteria are exemplary, and not exalts to, and, moreover, the

criteria correspond to Boolean logical expressions which may be implemented to perform the decision in step 614. If, in step 614 the certificate being processed is "better" the current certificate may be said to supercede a certificate in the composite keystore and in step 616 the preexisting certificate in the composite is replaced with the current certificate, and process 600 returns to step 608. Otherwise, if, in step 614 it is determined that current certificate is not "better" then the composite, step 614 proceeds by the "No" branch to step 608.

Returning to step 604, after all keystore sources have been looped over, step 604 proceeds by the "False" branch, and in step 618, the keystore object is returned. Security class methods may be called to access the composite keystore object.

Refer now to FIGURE 5 illustrating, in flow chart form, methodology 500 for secured data transfer using a distributed keystore in accordance with the principles of the present invention. In step 502, the distributed keystore is accessed. Access may be provided, via a plurality of protocols. For example, a connection to the keystore and certificate requests communicated in accordance with the file transport protocol (FTP), hypertext transfer protocol (HTTP) and the Secure Sockets Layer (SSL) extension thereof (HTTPs), or via a directory protocol such as the lightweight directory access protocol (LDAP). (These protocols have been standardized for use on the Internet, the specifications for which are published in corresponding Requests for Comments (RFC): RFC 414 (FTP), RFC 2616 (HTTP) and RFC 2251 (LDAP).) Additionally, conventional file access using a path name statement may also be provided in an embodiment in accordance with the principles of the present invention. Likewise, the keystore may be

identified by a similarly formatted string identifying the keystores location and the protocol for delivery of the certificates, commonly referred to as a Uniform Resource Locator (URL). (URLs are formatted in accordance with the syntax specified in RFC 1738. The portion of a URL specifying the access and transfer protocol, such as HTTP, and followed by ":", is referred to as the scheme specifier, and conventional file access, as previously described, may be effected by specifying "file" as the scheme in accordance with the URL syntax specification.)

The URLs for a keystore in accordance with the principles of the present invention may, for example in an embodiment implemented in a Java™ environment, be an instance of the Java™ KeyStore class. The distributed keystore may be specified by associating the name of the keystore with an instance of the keystore type. Properties may be associated with the keystore, which properties may be contained in a `java.security` class file. Instances of Java™ `Properties` objects may be associated with each URL which may be used to establish a connection to the keystore. An application requiring a certificate may retrieve a URL for the keystore by invoking the `getProperty` method of the `java.security` class. Step 502 may then be performed by establishing a connection to the keystore in accordance with the transport protocol associated with the scheme specified in the retrieved URL.

In step 504, a certificate including the public key needed to send the secure message is requested. The request may be formatted in accordance with the transport protocol specified in the scheme of the URL for the keystore, which URL may have been retrieved as previously described. Although the steps 502 and 504 have been discussed

in the context of a Java environment, an artisan of ordinary skill would recognize that the principles of the present invention may be embodied in other environments, and such embodiments would be within the spirit and scope of the present invention.

In step 506, it is determined if this requested certificate is returned. If a requested certificate is returned, then the requesting application encrypts the data to be transferred using the public key in the certificate, step 506. The encryption methodology may be in accordance with the encryption algorithm specified in the public key field 414, FIGURE 4. If however, the requested certificate is not in the distributed keystore, step 506 proceeds by the "No" branch, and the users local keystore is searched, step 510. If, the certificate is in the local keystore, step 512, then in step 514 the system administrator is notified to update the distributed keystore since the users local keystore and the distributed keystore are out of synchronization. The data to be transferred may then be encrypted using the recipient's public key obtained from the certificate retrieved from the users local keystore, step 508. If, however, in step 512 the requested certificate is not in the user's local keystore, then a secure transfer is unavailable, and the user notified, step 516.

Although the present invention and its advantages have been described in detail, it should be understood that various changes, substitutions and alterations can be made herein without departing from the spirit and scope of the invention as defined by the appended claims.